Calhoun: The NPS Institutional Archive

Faculty and Researcher Publications | Faculty and Researcher Publications

2009-10-29

Cryptographic Attacks and Countermeasures - A Mathematical View

Canright, David

Monterey, California. Naval Postgraduate School

http://hdl.handle.net/10945/37804

Calhoun is a project of the Dudley Knox Library at NPS, furthering the precepts and goals of open government and government transparency. All information contained herein has been approved for release by the NPS Public Affairs Officer.

Dudley Knox Library / Naval Postgraduate School
411 Dyer Road / 1 University Circle
Monterey, California USA 93943

http://www.nps.edu/library

Cryptographic Attacks and Countermeasures – A Mathematical View

David Canright & Pante Stanica

Naval Postgraduate School
Applied Mathematics Department,
Monterey, CA 93943, USA

1^{st} NPS Cyber Summit;
October 29, 2009

Basic Terminology

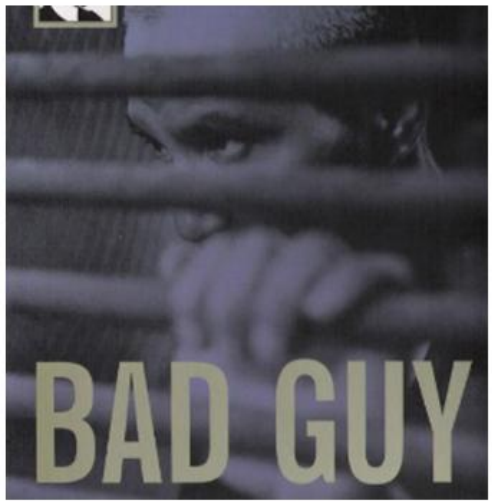
- **Cryptology**: All-inclusive term used for the study of secure communications over non-secure channels and related problems.
- **Cryptography**: The process of designing systems (ciphers, cryptosystems) to realize secure communications over non-secure channels.
- **Cryptoanalysis**: discipline of breaking the crypto systems.
- **Coding Theory**: Deals with representing the information using codes. It covers: compression, secrecy, and error-correction.
- Unencrypted data is called *plaintext* (cleartext); Encrypted data is called *ciphertext*
- *key*: string of digits that acts as a password. Only the intended receivers should have the key that transforms the ciphertext into plaintext.

Crypto setting

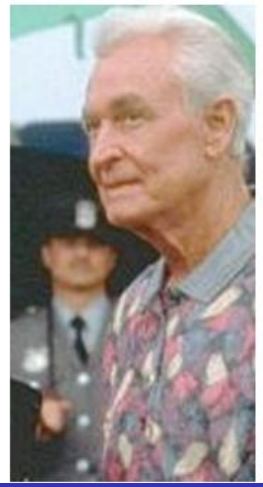
- Alice and Bob communicate;
- Eve (Oscar) listens

Alice

Eve (or Oscar)



Bob



Security Requirements

- **Confidentiality**: protection from disclosure to unauthorized persons
- **Integrity**: maintaining data consistency
- **Authentication**: assurance of identity of person or originator of data
- **Non-repudiation**: originator of communications cannot deny it later
- **Availability**: legitimate users have access when they need it
- **Access Control**: unauthorized users are kept out

Kerckhkoffs' Principle

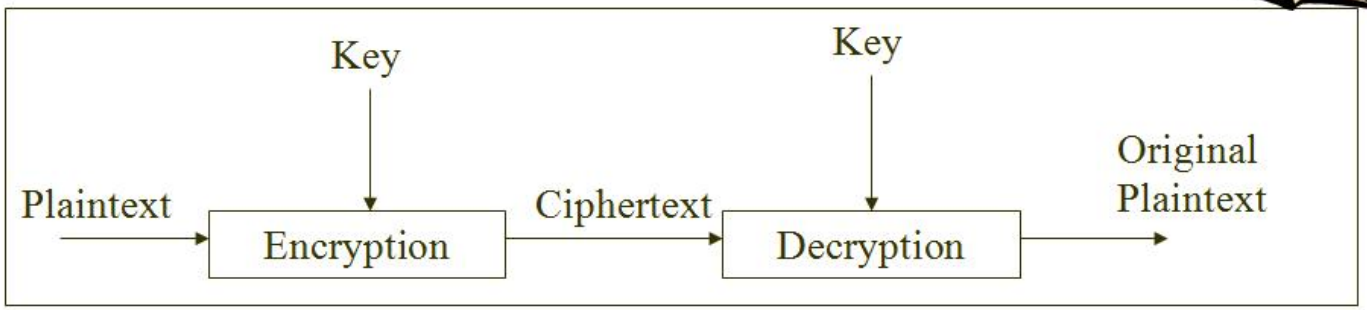
While assessing the strength of a cryptosystem, one should always assume that the enemy knows the cryptographic algorithm used.

- The security of the system, therefore, should be based (and analyzed) on:
  - the quality (strength) of the algorithm but not its obscurity
  - the key space (or key length)

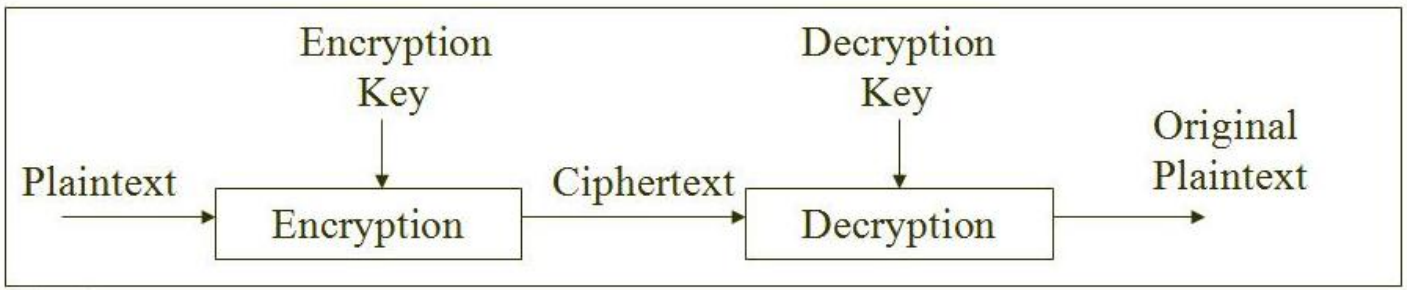
Mathematics comes into the play: "*(How) To encrypt or (how) not to encrypt, that is the question*"

Modern Cipher Systems

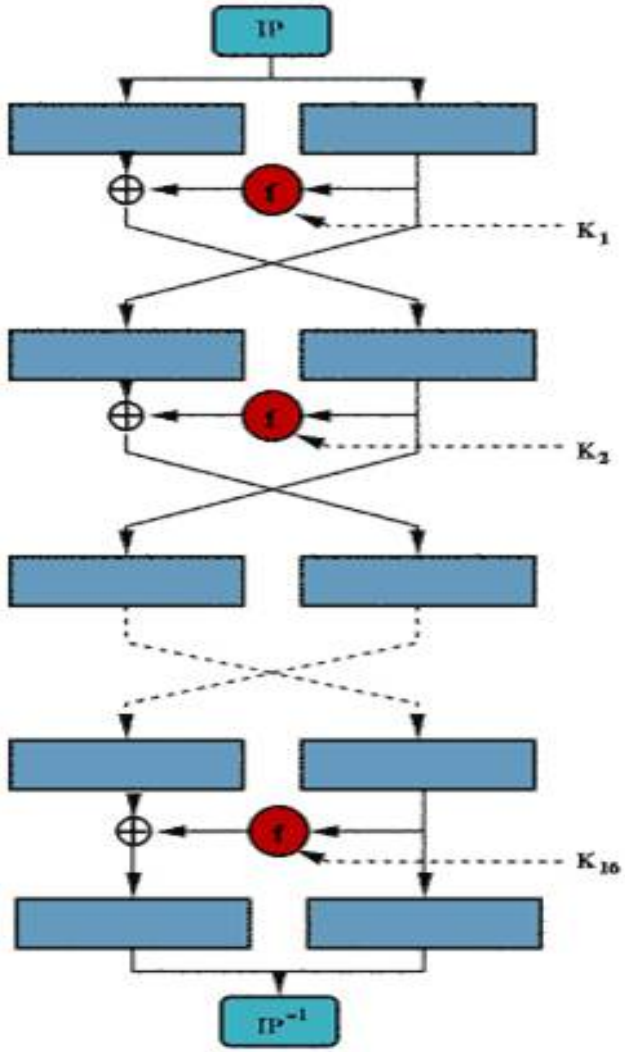
Symmetric Key Algorithms



Public Key Cryptography



Data Encryption Standard



Hey, where's the math in these... pictures?

- Every such cryptosystem can be described by a system of (nonlinear) equations

Hey, where's the math in these... pictures?

- Every such cryptosystem can be described by a system of (nonlinear) equations
- **Challenge**: Solve it/them, or show that it's... hard to do that.

New attack on Bluetooth E0

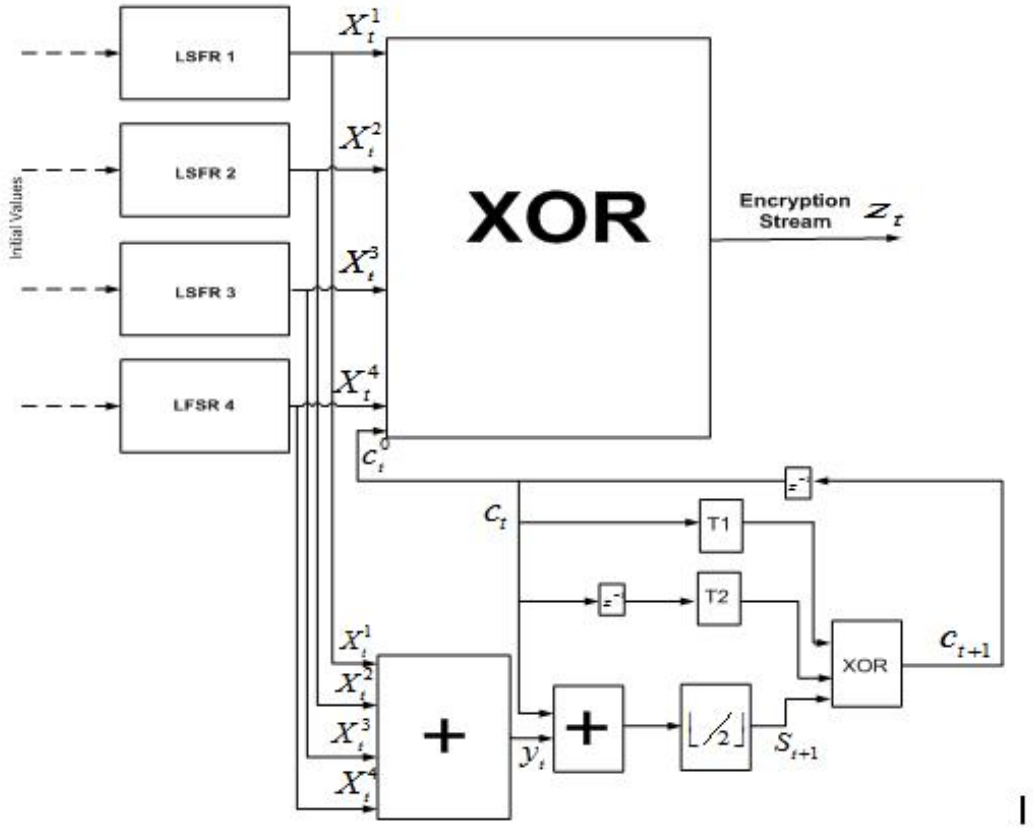


Figure 1. Encryption Procedure [after 39]

The four linear feedback shift registers E0 (LFSR1, LFSR2, LFSR and LFSR4) of E0 have the following lengths:

$$L_1 = 25, L_2 = 31, L_3 = 33, L_4 = 39.$$

Here's one equation describing E0

$$
\begin{aligned}
0 = {} & a \oplus b \oplus c \oplus d \oplus \\
& (x_5x_6 \oplus x_5x_7 \oplus x_5x_8 \oplus x_6x_7 \oplus x_6x_8 \oplus x_7x_8)(a \oplus b \oplus c \oplus d) \oplus \\
& x_5x_6x_7x_8 \oplus \\
& (x_5 \oplus x_6 \oplus x_7 \oplus x_8)(a \oplus b \oplus c \oplus d \oplus ab \oplus bc \oplus bd) \oplus \\
& x_1 \oplus x_2 \oplus x_3 \oplus x_4 \oplus \\
& (x_1 \oplus x_2 \oplus x_3 \oplus x_4)(x_5 \oplus x_6 \oplus x_7 \oplus x_8)(1 \oplus b) \oplus \\
& (x_1 \oplus x_2 \oplus x_3 \oplus x_4)(x_5x_6 \oplus x_5x_7 \oplus x_5x_8 \oplus x_6x_7 \oplus x_6x_8 \oplus x_7x_8) \oplus \\
& (x_9 \oplus x_{10} \oplus x_{11} \oplus x_{12})b \oplus \\
& (x_9 \oplus x_{10} \oplus x_{11} \oplus x_{12})(x_5 \oplus x_6 \oplus x_7 \oplus x_8)c(b \oplus 1) \oplus \\
& (x_9 \oplus x_{10} \oplus x_{11} \oplus x_{12})(x_5x_6 \oplus x_5x_7 \oplus x_5x_8 \oplus x_6x_7 \oplus x_6x_8 \oplus x_7x_8)c \oplus \\
& (x_9x_{10} \oplus x_9x_{11} \oplus x_9x_{12} \oplus x_{10}x_{11} \oplus x_{10}x_{12} \oplus x_{11}x_{12}) \oplus \\
& (x_9x_{10} \oplus x_9x_{11} \oplus x_9x_{12} \oplus x_{10}x_{11} \oplus x_{10}x_{12} \oplus x_{11}x_{12})(x_5 \oplus x_6 \oplus x_7 \oplus x_8)(1 \oplus b) \oplus \\
& (x_9x_{10} \oplus x_9x_{11} \oplus x_9x_{12} \oplus x_{10}x_{11} \oplus x_{10}x_{12} \oplus x_{11}x_{12})(x_5x_6 \oplus x_5x_7 \oplus x_5x_8 \oplus x_6x_7 \oplus x_6x_8 \oplus x_7x_8) \oplus \\
& x_{13} \oplus x_{14} \oplus x_{15} \oplus x_{16} \oplus \\
& (x_{13} \oplus x_{14} \oplus x_{15} \oplus x_{16})(x_5 \oplus x_6 \oplus x_7 \oplus x_8)(b \oplus 1) \oplus \\
& (x_{13} \oplus x_{14} \oplus x_{15} \oplus x_{16})(x_5x_6 \oplus x_5x_7 \oplus x_5x_8 \oplus x_6x_7 \oplus x_6x_8 \oplus x_7x_8).
\end{aligned}
$$

Attacks on E0

- Armknecht and Krause [2003] - *algebraic attack* on Bluetooth E0; 2^{69} operations
- Lu and Vaudenay [2004] - *correlation attack*; 2^{37} operations
- Petrakos, Dinolt, Michael and Stanica [2009] *cube attack*; 2^{21} operations
- Cube attack: Proposed by Dinur & Shamir (2008)
- Thesis Research Topics: apply the cube attack to ECRYPT eStream project ciphers (like GRAIN-128)

All cryptosystems based on a "difficult" mathematical part...

- Shannon (*Communication theory of secrecy systems*, Bell Systems Tech. J. 28, (1949), 656–715) proposed two criteria:
  * *Diffusion*: one character change in the plaintext should effect as many ciphertext characters as possible;
  * *Confusion*: The key should not relate to the ciphertext in a "simple" way.

All cryptosystems based on a "difficult" mathematical part...

- Shannon (*Communication theory of secrecy systems*, Bell Systems Tech. J. 28, (1949), 656–715) proposed two criteria:
  * *Diffusion*: one character change in the plaintext should effect as many ciphertext characters as possible;
  * *Confusion*: The key should not relate to the ciphertext in a "simple" way.
- Impose some criteria to S-boxes or combiners to respect these principles: balancedness, high avalanche features, high nonlinearity, correlation immunity, algebraic immunity, etc.

All cryptosystems based on a "difficult" mathematical part...

- Shannon (*Communication theory of secrecy systems*, Bell Systems Tech. J. 28, (1949), 656–715) proposed two criteria:
  * *Diffusion*: one character change in the plaintext should effect as many ciphertext characters as possible;
  * *Confusion*: The key should not relate to the ciphertext in a "simple" way.
- Impose some criteria to S-boxes or combiners to respect these principles: balancedness, high avalanche features, high nonlinearity, correlation immunity, algebraic immunity, etc.
- Unfortunately, there are trade-offs among these criteria!

Crypto properties of S-boxes

- Can one find large classes of Boolean functions that satisfy one and hopefully more of these criteria?

Crypto properties of S-boxes

- Can one find large classes of Boolean functions that satisfy one and hopefully more of these criteria?
- Collaborations between researchers in various areas is warranted: I mention my collaboration with Jon T. Butler (ECE) who's an expert in reconfigurable computing (SRC-6), to enumerate and find Boolean functions with good "crypto" properties;

Crypto properties of S-boxes

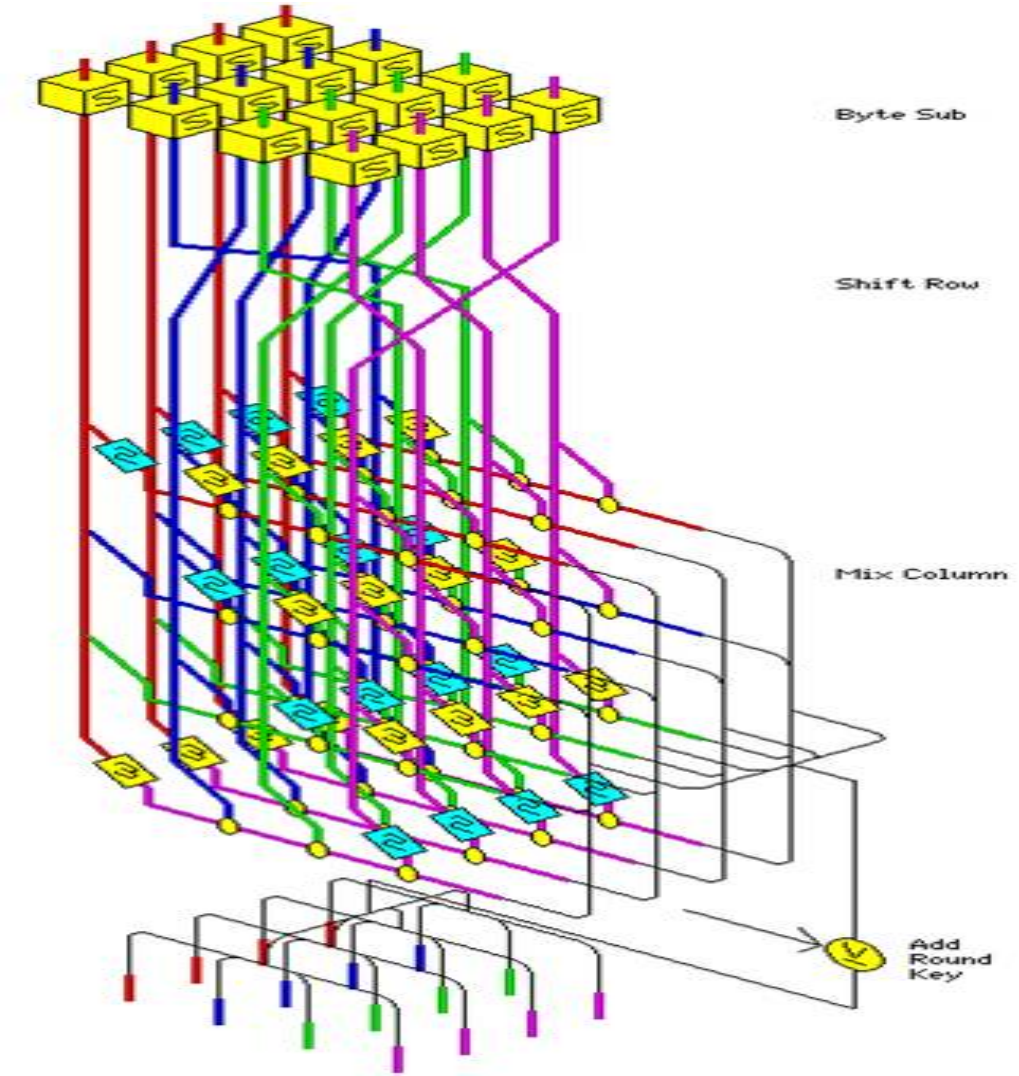
- Can one find large classes of Boolean functions that satisfy one and hopefully more of these criteria?
- Collaborations between researchers in various areas is warranted: I mention my collaboration with Jon T. Butler (ECE) who's an expert in reconfigurable computing (SRC-6), to enumerate and find Boolean functions with good "crypto" properties;
- How hard can it be to perform the search through the entire space?

FPGA vc. PC Computational Complexity

n	# of Functions	Comp. Time – All Functions 100 MHz. = *A*	Comp. Time – Conventional PC = *B*	Speedup *B*/*A*
2	16	0.16 μsec.	6.38 μsec.	40×
3	256	2.56 μsec.	457 μsec.	179×
4	65,536	655.4 μsec.	0.388 sec.	592×
5	4.2950×10^{9}	42.9 sec.	25.338 hrs.	2,126×
6	1.8447×10^{19}	5,849 yrs.	3.98×10^{7} yrs.	6,506×
7	3.4028×10^{38}	1.1×10^{23} yrs.	--	--
8	1.1579×10^{77}	3.7×10^{61} yrs.	--	--
9	1.3408×10^{154}	4.3×10^{138} yrs.	--	--

This shows the speedup obtained through the use of the SRC-6's FPGA compared to an implementation using the 2.8 GHz. Xeon microprocessor on the SRC-6. -- data is not available

Advanced Encryption Standard



Masking AES

- *"Side-channel attacks"* can compromise implementations of cryptographic algorithms, including AES.
- Such attacks use *statistics* of phenomena such as *timing*, *power*, or *electromagnetic radiation*, to infer the secret key.
- One countermeasure is to add random data at each step, to *randomize* these statistics; this is called *masking*.
- Here is a glimpse of the mathemagic that makes it work.

AES Algorithm

- AES is a symmetric 128-bit block cipher:
- from 128, 192, or 256 bit key, generate a different 128-bit round key for each of 10, 12, or 14 rounds;
- process each block by rounds:

round 0 :	Add Round Key.
1 to $n-1$:	Substitute Bytes;
	Shift Rows;
	Mix Columns;
	Add Round Key.
round n :	Substitute Bytes;
	Shift Rows;
	Add Round Key.

Nonlinearity/Complexity

of the four steps:

- the steps Shift Rows, Mix Columns, & Add Round Key are *linear* operations (and easy)
- the Substitute Bytes function is nonlinear due to the *inverse* operation in the 8-bit field, and is complicated to compute.

Nonlinearity/Complexity

of the four steps:

- the steps Shift Rows, Mix Columns, & Add Round Key are *linear* operations (and easy)
- the Substitute Bytes function is nonlinear due to the *inverse* operation in the 8-bit field, and is complicated to compute. The Substitute Bytes step itself comprises two substeps; for each 8-bit byte $\mathbf{a}$:
  1. *Inverse*: Find $\mathbf{c} = \mathbf{a}^{-1}$, the inverse in the 8-bit field;
  2. *Affine*: Then the output is $\mathbf{s} = M\,\mathbf{c} \oplus \mathbf{b}$.

field: definition

A *field* is a set of numbers with the operations of addition, subtraction, multiplication, and division, with all the usual properties. Familiar examples:

- Rational numbers
- Real numbers
- Complex numbers

field: definition

A *field* is a set of numbers with the operations of addition, subtraction, multiplication, and division, with all the usual properties. Familiar examples:

- Rational numbers
- Real numbers
- Complex numbers

Galois showed that one can also have fields with a *finite* number of elements. Some examples:

- modulo 2: $\{0, 1\}$ where $1 \oplus 1 = 0$
- field of 8-bit bytes, as used in AES

Subfield Representation

The trick that makes masking work nicely is using subfields...

- *standard*
  - for the 8-bit field: $\mathbf{A} = a_7 x^7 + \cdots + a_1 x + a_0$, where each a_i is one bit and $x^8 + x^4 + x^3 + x + 1 = 0$.

Subfield Representation

The trick that makes masking work nicely is using subfields...

- *standard*
  - for the 8-bit field: $\mathbf{A} = a_7 x^7 + \cdots + a_1 x + a_0$, where each a_i is one bit and $x^8 + x^4 + x^3 + x + 1 = 0$.
- *subfield*
  - for the 8-bit field: $\mathbf{A} = A_1 x + A_0$, where A_i are in the 4-bit field and $x^2 + x + 8 = 0$;

Notation: $\mathbf{A}$ in the 8-bit field; A in the 4-bit field; $\mathbf{a}$ in the 2-bit field; a in the 1-bit field

Subfield Representation

The trick that makes masking work nicely is using subfields...

- *standard*
  - for the 8-bit field: $\mathbf{A} = a_7 x^7 + \cdots + a_1 x + a_0$, where each a_i is one bit and $x^8 + x^4 + x^3 + x + 1 = 0$.
- *subfield*
  - for the 8-bit field: $\mathbf{A} = A_1 x + A_0$, where A_i are in the 4-bit field and $x^2 + x + 8 = 0$;
  - then for the 4-bit field: $A = \mathbf{a}_1 x + \mathbf{a}_0$, where $\mathbf{a}_i$ are in the 2-bit field and $x^2 + x + 3 = 0$;
  - then for the 2-bit field: $\mathbf{a} = a_1 x + a_0$, where a_i are in the 1-bit field and $x^2 + x + 1 = 0$.

Notation: $\mathbf{A}$ in the 8-bit field; A in the 4-bit field; $\mathbf{a}$ in the 2-bit field; a in the 1-bit field

Inversion using subfields

without masking

- Inversion in the 8-bit field involves one inversion and several multiplications and additions in the 4-bit field.
- Inversion in the 4-bit field involves one inversion and several multiplications and additions in the 2-bit field.

Inversion using subfields

without masking

- Inversion in the 8-bit field involves one inversion and several multiplications and additions in the 4-bit field.
- Inversion in the 4-bit field involves one inversion and several multiplications and additions in the 2-bit field.
- Inversion in the 2-bit field is just a bit swap:

  Given: $\mathbf{c} = [c_0\,,\, c_1]$

  Result: $\mathbf{c}^{-1} = [c_1\,,\, c_0]$

Inversion using subfields

without masking

- Inversion in the 8-bit field involves one inversion and several multiplications and additions in the 4-bit field.
- Inversion in the 4-bit field involves one inversion and several multiplications and additions in the 2-bit field.
- Inversion in the 2-bit field is just a bit swap:

  Given: $\mathbf{c} = [c_0\,,\, c_1]$

  Result: $\mathbf{c}^{-1} = [c_1\,,\, c_0]$

  Note: Inversion in the 2-bit field is linear

Masked Inversion

1. add a random mask in the 8-bit field
2. in the 4-bit field and the 2-bit field, add mask "corrections" for the multiplications
3. for the inversion in the 2-bit field, just swap the two bits of the mask

Then all operations remain masked and *every intermediate result is statistically random!*

Algebraic Attacks on AES

AES is defined by relatively simple algebraic equations in finite fields. Consequently, there has been much research in ways to solve the system of nonlinear equations, to find the secret key.

- Courtois et al. attacked small versions of AES, using equations in the field of bits. The method involved doing linear algebra, but on the nonlinear terms.
- Cid et al. also attacked small versions of AES, but using many fewer equations, in the field of bytes. The method applied standard tools for solving polynomial equations.
- Raddum and Semaev developed a whole new way of representing nonlinear equations over finite fields, called MRHS equations, which was orders of magnitude faster, for certain special cases.
- Our current work explores ways to make this new approach more efficient...

What else is there to increase the complexity of cryptosystems?

- Hmmm... invent other systems!
- Working with arithmetic on algebraic curves ▸ Go;

What else is there to increase the complexity of cryptosystems?

- Hmmm... invent other systems!
- Working with arithmetic on algebraic curves Go;
- Working in a world where $a \cdot b$ is not the same as $b \cdot a$ (non-commutative structures);

What else is there to increase the complexity of cryptosystems?

- Hmmm... invent other systems!
- Working with arithmetic on algebraic curves Go;
- Working in a world where $a \cdot b$ is not the same as $b \cdot a$ (non-commutative structures);
- Quantum environments.

somewhat

- Alice and Bob are able to talk securely!

Theorem (David Canright & Pante Stanica)

Thank you for your attention.

Proof.

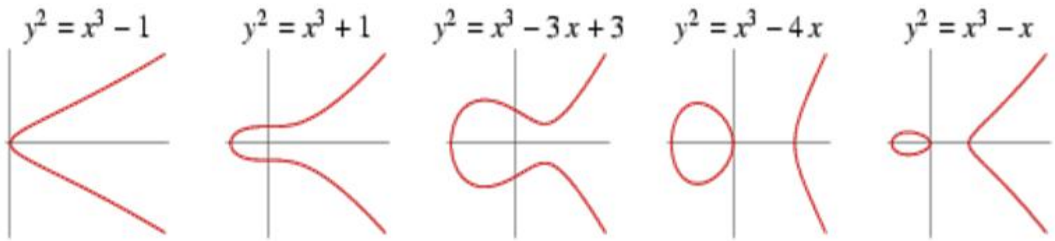
No proof required! □

General form of an EC

- An *elliptic curve* is a plane curve defined by an equation of the form

$$y^2 = x^3 + ax + b$$

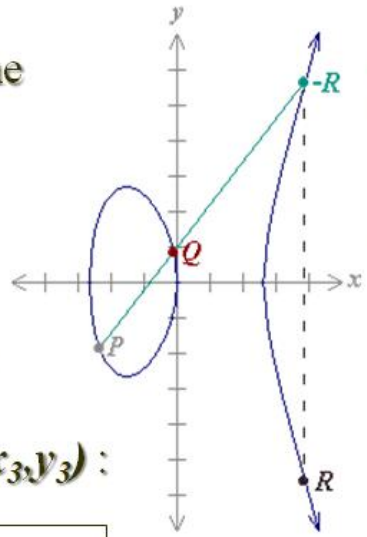
Examples



EC group - sum of two points

Define for two points $P(x_1,y_1)$ and $Q(x_2,y_2)$ in the elliptic curve

$$\lambda = \begin{cases} \dfrac{y_2 - y_1}{x_2 - x_1} \text{ for } x_1 \neq x_2 \\ \dfrac{3x_1^2 + a}{2y_1} \text{ for } x_1 = x_2 \end{cases}$$



P (-2.35, -1.86)
Q (-0.1, 0.836)
-R (3.89, 5.62)
R (3.89, -5.62)

P + Q = R = (3.89, -5.62).

Then $P+Q$ is given by $R(x_3,y_3)$:

$$\boxed{\begin{aligned} x_3 &= \lambda - x_1 - x_2 \\ y_3 &= \lambda(x_3 - x_1) + y_1 \end{aligned}}$$

$y^2 = x^3 - 7x$

Example – Elliptic Curve Cryptosystem

- Alice wants to send Bob an encrypted message M.
  - Both agree on a base point, B.
  - Alice and Bob create public/private keys.
    - **Alice**
      - **Private Key = a integer**
      - **Public Key = P_A = a * B**
    - **Bob**
      - **Private Key = b integer**
      - **Public Key = P_B = b * B**
  - Alice takes the plaintext M and encodes it onto a point, P_M, from the elliptic group

Example – Elliptic Curve Cryptosystem

- Alice chooses another random integer, k from the interval [1, p-1]
- The ciphertext is a pair of points
  - $P_C = [\,(kB), (P_M + kP_B)\,]$

- To decrypt, Bob computes the product of the first point from P_C and his private key, b
  - $b * (kB)$
- Bob then takes this product and subtracts it from the second point from P_C
  - $(P_M + kP_B) - [b(kB)] = P_M + k(bB) - b(kB) = P_M$
- Bob then decodes P_M to get the message, M.